Perception Processing for General Intelligence: Bridging the Symbolic/Subsymbolic Gap

Ben Goertzel

Novamente LLC

Abstract. Bridging the gap between symbolic and subsymbolic representations is a – perhaps *the* – key obstacle along the path from the present state of AI achievement to human-level artificial general intelligence. One approach to bridging this gap is hybridization – for instance, incorporation of a subsymbolic system and a symbolic system into a integrative cognitive architecture. Here we present a detailed design for an implementation of this approach, via integrating a version of the DeSTIN deep learning system into OpenCog, an integrative cognitive architecture including rich symbolic capabilities. This is a "tight" integration, in which the symbolic and subsymbolic aspects exert detailed real-time influence on each others' operations. An earlier technical report has described in detail the revisions to DeSTIN needed to support this integration, which are mainly along the lines of making it more "representationally transparent," so that its internal states are easier for OpenCog to understand.

1 Introduction

While it's widely accepted that human beings carry out both *symbolic* and *subsymbolic* processing, as integral parts of their general intelligence, the precise definition of "symbolic" versus "subsymbolic" is a subtle issue, which different AI researchers will approach in different ways depending on their differing overall perspectives on AI. Nevertheless, the intuitive meaning of the concepts is commonly understood:

– **"subsymbolic"** refers to things like pattern recognition in high-dimensional quantitative sensory data, and real-time coordination of multiple actuators taking multidimensional control signals
– **"symbolic"** refers to things like natural language grammar and (certain or uncertain) logical reasoning, that are naturally modeled in terms of manipulation of symbolic tokens in terms of particular (perhaps experientially learned) rules

Views on the relationship between these two aspects of intelligence in human and artificial cognition are quite diverse, including perspectives such as

1. Symbolic representation and reasoning are the core of human-level intelligence; subsymbolic aspects of intelligence are of secondary importance and

can be thought of as pre or post processors to symbolic representation and reasoning

2. Subsymbolic representation and learning are the core of human intelligence; symbolic aspects of intelligence
   (a) emerge from the subsymbolic aspects as needed; or,
   (b) arise via a relatively simple, thin layer on top of subsymbolic intelligence, that merely applies subsymbolic intelligence in a slightly different way
3. Symbolic and subsymbolic aspects of intelligence are best considered as different subsystems, which
   (a) have a significant degree of independent operation, but also need to coordinate closely together; or,
   (b) operate largely separately and can be mostly considered as discrete modules

In evolutionary terms, it is clear that subsymbolic intelligence came first, and that most of the human brain is concerned with the subsymbolic intelligence that humans share with other animals. However, this observation doesn't have clear implications regarding the relationship between symbolic and subsymbolic intelligence in the context of everyday cognition.

In the history of the AI field, the symbolic/subsymbolic distinction was sometimes aligned with the dichotomy between logic-based and rule-based AI systems (on the symbolic side) and neural networks (on the subsymbolic side) [1]. However, this dichotomy has become much blurrier in the last couple decades, with developments such as neural network models of language parsing [2] and logical reasoning [3], and symbolic approaches to perception and action [4]. Integrative approaches have also become more common, with one of the major traditional symbolic AI systems, ACT-R, spawning a neural network version [5] with parallel structures and dynamics to the traditional explicitly symbolic version and a hybridization with a computational neuroscience model [6]; and another one, SOAR, incorporating perception processing components as separate modules [7]. The field of "neural-symbolic computing" has emerged, covering the emergence of symbolic rules from neural networks, and the hybridization of neural networks with explicitly symbolic systems [8].

Our goal here is not to explore the numerous deep issues involved with the symbolic/subsymbolic dichotomy, but rather to describe the details of a particular approach to symbolic/subsymbolic integration, inspired by Perspective 3a in the above list: the consideration of symbolic and subsymbolic aspects of intelligence as different subsystems, which have a significant degree of independent operation, but also need to coordinate closely together. We believe this kind of integration can serve a key role in the quest to create human-level general intelligence. The approach presented here is at the beginning rather than end of its practical implementation; what we are describing here is the initial design intention of a project in progress, which is sure to be revised in some respects as implementation and testing proceed. We will focus mainly on the tight integration of a subsymbolic system enabling gray-scale vision processing into a cognitive architecture with significant symbolic aspects. A longer version of the paper, available online [9], explains how the same ideas can be used for color vision, and multi-sensory and perception-action integration.

The approach presented here begins with two separate AI systems, both currently implemented in open-source software:

- **OpenCog**, an integrative architecture for AGI [10] [11], which is centered on a "weighted, labeled hypergraph" knowledge representation called the Atomspace, and features a number of different, sophisticated cognitive algorithms acting on the Atomspace. Some of these cognitive algorithms are heavily symbolic in focus (e.g. a probabilistic logic engine); others are more subsymbolic in nature (e.g. a neural net like system for allocating attention and assigning credit). However, OpenCog in its current form cannot deal with high-dimensional perceptual input, nor with detailed real-time control of complex actuators. OpenCog is now being used to control intelligent characters in an experimental virtual world, where the perceptual inputs are the 3D coordinate locations of objects or small blocks; and the actions are movement commands like "step forward", "turn head to the right" [12] [13]. OpenCog is an open-source AGI software framework, which has been used for various practical applications in the area of natural language processing and data mining; e.g. see [14], and also for the in-progress implementation of the OpenCogPrime design aimed ultimately toward AGI at the human level and beyond.
- **DeSTIN** [15],[16], a deep learning system consisting of a hierarchy of processing nodes, in which the nodes on higher levels correspond to larger regions of space-time, and each node carries out prediction regarding events in the space-time region to which it corresponds. Feedback and feedforward dynamics between nodes combine with the predictive activity within nodes, to create a complex nonlinear dynamical system whose state self-organizes to reflect the state of the world being perceived. The core concepts of DeSTIN are similar to those of Jeff Hawkins' Numenta system [17] [18], Dileep George's work (`http://vicariousinc.com`) and work by Mohamad Tarifi [19], Bundzel and Hashimoto [20], and others. In the terminology introduced in [21], DeSTIN is an example of a Compositional Spatiotemporal Deep Learning System, or CSDLN. However, compared to other CSDLNs, the specifics of DeSTIN's dynamics have been designed in what we consider a particularly powerful way, and the system has shown good results on small-scale test problems [22]. So far DeSTIN has been utilized only for vision processing, but a similar proprietary system has been used for auditory data as well; and DeSTIN was designed to work together with an accompanying action hierarchy.

We will not review particulars of OpenCog nor DeSTIN here, referring the reader to the above-cited references, and assuming basic knowledge of how both systems work. These two systems were not originally designed to work together, but we will describe a method for achieving their tight integration via

1. Modifying DeSTIN in several ways, so that

   (a) the patterns in its states over time will have more easily recognizable regularities
   (b) its nodes are able to scan their inputs not only for simple statistical patterns (DeSTIN "centroids"), but also for patterns recognized by routines supplied to it by an external source (e.g. another AI system such as OpenCog)
2. Utilizing one of OpenCog's cognitive processes (the "Fishgram" frequent subhypergraph mining algorithm) to recognize patterns in sets of DeSTIN states, and then recording these patterns in OpenCog's Atomspace knowledge store
3. Utilizing OpenCog's other cognitive processes to abstract concepts and draw conclusions from the patterns recognized in DeSTIN states by Fishgram
4. Exporting the concepts and conclusions thus formed to DeSTIN, so that its nodes can explicitly scan for their presence in their inputs, thus allowing the results of symbolic cognition to explicitly guide subsymbolic perception
5. As described in the the extended online version of the paper [9]: Creating an action hierarchy corresponding closely to DeSTIN's perceptual hierarchy, and also corresponding to the actuators of a particular robot. This allows action learning to be done via an optimization approach ([23], [24]), where the optimization algorithm uses DeSTIN states corresponding to perceived actuator states as part of its inputs.

The ideas described here have mostly not yet been implemented, but work has begun on Items 1a (modifying DeSTIN so that the patterns in its states over time will have more easily recognizable regularities) and 2 (utilizing Fishgram to recognize patterns in DeSTIN system states), as part of a 2012 Google Summer of Code project. Item 1a has been covered in the technical report [25]; the remainder of the points are discussed here.

The ideas presented here are compatible with those described in [21], but different in emphasis. That paper described a strategy for integrating OpenCog and DeSTIN via creating an intermediate "semantic CSDLN" hierarchy to translate between OpenCog and DeSTIN, in both directions. In the approach suggested here, this semantic CSDLN hierarchy exists conceptually but not as a separate software object: it exists as the combination of

– OpenCog predicates exported to DeSTIN and used alongside DeSTIN centroids, inside DeSTIN nodes
– OpenCog predicates living in the OpenCog knowledge repository (AtomSpace), and interconnected in a hierarchical way using OpenCog nodes and links (thus reflecting DeSTIN's hierarchical structure within the AtomSpace).

This hierarchical network of predicates, spanning the two software systems, plays the role of a semantic CSDLN as described in [21].

Simplified OpenCog Workflow The dynamics inside an OpenCog system may be highly complex, defying simple flowcharting, but from the point of view of OpenCog-DeSTIN integration, one important pattern of information flow through the system is as follows:

1. Perceptions come into the Atomspace. In the current OpenCog system, these are provided via a proxy to the game engine where the OpenCog controlled character interacts. In an OpenCog-DeSTIN hybrid, these will be provided via DeSTIN.
2. Hebbian learning builds HebbianLinks between perceptual Atoms representing percepts that have frequently co-occurred
3. PLN inference, concept blending and other methods act on these perceptual Atoms and their HebbianLinks, forming links between them and linking them to other Atoms stored in the Atomspace reflecting prior experience and generalizations therefrom
4. Attention allocation gives higher short and long term importance values to those Atoms that appear likely to be useful based on the links they have obtained
5. Based on the system's current goals and subgoals (the latter learned from the top-level goals using PLN), and the goal-related links in the Atomspace, the OpenPsi mechanism triggers the PLN-based planner, which chooses a series of high-level actions that are judged likely to help the system achieve its goals in the current context
6. The chosen high-level actions are transformed into series of lower-level, directly executable actions. In the current OpenCog system, this is done by a set of hand-coded rules based on the specific mechanics of the game engine where the OpenCog controlled character interacts. In an OpenCog-DeSTIN hybrid, the lower-level action sequence will be chosen by an optimization method acting based on the motor control and perceptual hierarchies.

2 Integrating DeSTIN and OpenCog

The integration of DeSTIN and OpenCog involves two key aspects:

– recognition of patterns in sets of DeSTIN states, and exportation of these patterns into the OpenCog Atomspace
– use of OpenCog-created concepts within DeSTIN nodes, alongside statistically-derived "centroids"

From here on, unless specified otherwise, when we mention "DeSTIN" we will refer to "Uniform DeSTIN" as defined in the technical report [25], an extension of "classic DeSTIN" as defined in [15]. The essential difference is that in Uniform DeSTIN, the same centroids are shared across the different nodes in the network; and, a belief can be matched with a centroid even if the two differ by some rotation or shear. So, in Uniform DeSTIN, each node compares its inputs to a library of known patterns in a manner that incorporates invariance to location, scale, rotation and shear.

2.1 Mining Patterns from DeSTIN States

The first step toward using OpenCog tools to mine patterns from sets of DeSTIN states, is to represent these states in Atom form in an appropriate way. A simple but workable approach, restricting attention for the moment to purely spatial patterns, is to use the six predicates: $hasCentroid(node\ N, int\ k)$, $hasParentCentroid(node\ N, int\ k)$, $hasNorthNeighborCentroid(node\ N, int\ k)$, $hasSouthNeighborCentroid(node\ N, int\ k)$, $hasEastNeighborCentroid(node\ N, int\ k)$, $hasWestNeighborCentroid(node\ N, int\ k)$. For instance, $hasNorthNeighborCentroid(N, 3)$ means that N's north neighbor has centroid #3. One may consider also the predicates: $hasParent(node\ N, Node\ M)$, $hasNorthNeighbor(node\ N, Node\ M)$, $hasSouthNeighbor(node\ N, Node\ M)$, $hasEastNeighbor(node\ N, Node\ M)$, $hasWestNeighbor(node\ N, Node\ M)$.

Now suppose we have a stored set of DeSTIN states, saved from the application of DeSTIN to multiple different inputs. What we want to find are predicates P that are *conjunctions* of instances of the above 10 predicates, which occur frequently in the stored set of DeSTIN states. A simple example of such a predicate would be the conjunction of

– $hasNorthNeighbor(\$N, \$M)$
– $hasParentCentroid(\$N, 5)$
– $hasParentCentroid(\$M, 5)$
– $hasNorthNeighborCentroid(\$N, 6)$
– $hasWestNeighborCentroid(\$M, 4)$

This predicate could be evaluated at any pair of nodes $(\$N, \$M)$ on the same DeSTIN level. If it is true for atypically many of these pairs, then it's a "frequent pattern", and should be detected and stored.

OpenCog's pattern mining component, Fishgram, exists precisely for the purpose of mining this sort of conjunction from sets of relationships that are stored in the Atomspace. It may be applied to this problem as follows:

– Translate each DeSTIN state into a set of relationships drawn from: hasNorthNeighbor, hasSouthNeighbor, hasEastNeighbor, hasWestNeighbor, hasCentroid, hasParent
– Import these relationships, describing each DeSTIN state, into the OpenCog Atomspace
– Run pattern mining on this AtomSpace.

2.2 Probabilistic Inference on Mined Hypergraphs

Patterns mined from DeSTIN states can then be reasoned on by OpenCog's PLN inference engine, allowing analogy and generalization.

Suppose centroids 5 and 617 are estimated to be similar – either via DeSTIN's built-in similarity metric, or, more interestingly via OpenCog inference on the Atom representations of these centroids. As an example of the latter, consider: 5 could represent a person's nose and 617 could represent a rabbit's nose. In this case, DeSTIN might not judge the two centroids particularly similar on a purely visual level, but, OpenCog may know that the images corresponding to both of these centroids are are called "noses" (e.g. perhaps via noticing people indicate these images in association with the word "nose"), and may thus infer (using a simple chain of PLN inferences) that these centroids seem probabilistically similar.

If 5 and 617 are estimated to be similar, then a predicate like

```
ANDLink
   EvaluationLink
      hasNorthNeighbor
      ListLink $N  $M
   EvaluationLink
      hasParentCentroid
      ListLink $N  5
   EvaluationLink
      hasParentCentroid
      ListLink $M  5
   EvaluationLink
      hasNorthNeighborCentroid
      ListLink $N 6
   EvaluationLink
      hasWestNeighborCentroid
      ListLink $M 4
```

mined from DeSTIN states, could be extended via PLN analogical reasoning to

```
ANDLink
   EvaluationLink
      hasNorthNeighbor
      ListLink $N $M
   EvaluationLink
      hasParentCentroid
      ListLink $N  617
   EvaluationLink
      hasParentCentroid
      ListLink $M  617
   EvaluationLink
      hasNorthNeighborCentroid
      ListLink $N 6
   EvaluationLink
      hasWestNeighborCentroid
      ListLink $M 4
```

2.3 Insertion of OpenCog-Learned Predicates into DeSTIN's Pattern Library

Suppose one has used Fishgram, as described above, to recognize predicates embodying frequent or surprising patterns in a set of DeSTIN states or state-sequences. The next natural step is to add these frequent or surprising patterns to DeSTIN's pattern library, so that the pattern library contains not only classic DeSTIN centroids, but also these corresponding ”image grammar” style patterns. Then, when a new input comes into a DeSTIN node, in addition to being compared to the centroids at the node, it can be fed as input to the predicates associated with the node.

What is the advantage of this approach, compared to DeSTIN without these predicates? The capability for more compact representation of a variety of spatial patterns. In many cases, a spatial pattern that would require a large number of DeSTIN centroids to represent, can be represented by a single, fairly compact predicate. It is an open question whether these sorts of predicates are really critical for human-like vision processing. However, our intuition is that they do have a role in human as well s machine vision. In essence, DeSTIN is based on a fancy version of nearest-neighbor search, applied in a clever way on multiple levels of a hierarchy, using context-savvy probabilities to bias the matching. But we suspect there are many visual patterns that are more compactly and intuitively represented using a more flexible language, such as OpenCog predicates formed by combining elementary predicates involving appropriate spatial and temporal relations.

For example, consider the archetypal spatial pattern of a face as: either two eyes that are next to each other, or sunglasses, above a nose, which is in turn above a mouth. (This is an oversimplified toy example, but we're positing it for illustration only. The same point applies to more complex and realistic patterns.) One could represent this in OpenCog's Atom language as something like:

```
AND
  InheritanceLink N B_nose
  InheritanceLink M B_mouth
  EvaluationLink
      above
      ListLink E N
  EvaluationLink
     above
     ListLink N M
  OR
    AND
       MemberLink E1  E
       MemberLink E2  E
       EvaluationLink
          next_to
          ListLink E1 E2
       InheritanceLink E1 B_eye
    AND
       InheritanceLink E  B_sunglasses
```

where e.g. *B_eye* is a DeSTIN belief that corresponds roughly to recognition of the spatial pattern of a human eye. To represent this using ordinary DeSTIN centroids, one couldn't represent the OR explicitly; instead one would need to split it into two different sets of centroids, corresponding to the eye case and the sunglasses case unless the DeSTIN pattern library contained a belief corresponding to "eyes or sunglasses." But the question then becomes: how would classic DeSTIN actually learn a belief like this? In the suggested architecture, pattern mining on the database of DeSTIN states is proposed as an algorithm for learning such beliefs.

This sort of predicate-enhanced DeSTIN will have advantages over the traditional version, only if the actual distribution of images observed by the system contains many (reasonably high probability) images modeled accurately by predicates involving disjunctions and/or negations as well as conjunctions. If the system's perceived world is simpler than this, then good old DeSTIN will work just as well, and the OpenCog-learned predicates are a needless complication.

3 Conclusion

We have described, at a high level, a novel approach to bridging the symbolic / subsymbolic gap, via very tightly integrating DeSTIN with OpenCog. We don't claim that this is the only way to bridge the gap, but we do believe it is a viable way. And while we have focused on robotics applications here, the basic ideas described could be implemented and evaluated in a variety of other contexts as well, for example the identification of objects and events in videos, or intelligent video summarization.

Our hope is that the hybridization of OpenCog and DeSTIN as described here will constitute a major step along the path to human-level AGI. It will enable the creation of an OpenCog instance endowed with the capability of flexibly interacting with a rich stream of data from the everyday human world. This data will not only help OpenCog to guide a robot in carrying out everyday tasks, but will also provide raw material for OpenCog's cognitive processes to generalize from in various ways – e.g. to use as the basis for the formation of new concepts or analogical inferences.

References

1. Pinker, S., JacquesMehler: Connections and Symbols. MIT Press (1988)
2. Garg, N., Henderson, J.: Temporal restricted boltzmann machines for dependency parsing. In: Proc. ACL. (2011)
3. Lehmann, J., Bader, S., Hitzler, P.: Extracting reduced logic programs from artificial neural networks. Applied Intelligence (2010)
4. Shanahan, M., Randell, D.A.: A logic-based formulation of active visual perception. In: Knowledge Representation. (2004)
5. Lebiere, C., Anderson, J.R.: A connectionist implementation of the act-r production system. In: Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society. (1993)
6. Jilk, D.J., Lebiere, C.: and o'reilly. R. C. and Anderson, J. R. (2008). SAL: An explicitly pluralistic cognitive architecture. Journal of Experimental and Theoretical Artificial Intelligence **20** (2008) 197–218
7. Laird, J.E.: The Soar Cognitive Architecture. MIT Press (2012)
8. Hammer, B., Hitzler, P., eds.: Perspectives of Neural-Symbolic Integration. Studies in Computational Intelligence, Vol. 77. Springer (2007)

9. Goertzel, B.: Perception processing for general intelligence: Bridging the symbolic/subsymbolic gap. (http://wp.goertzel.org/?p=404)
10. Goertzel, B.e.a.: Opencogbot: An integrative architecture for embodied agi. Proc. of ICAI-10, Beijing (2010)
11. Goertzel, B., Pitt, J., Wigmore, J., Geisweiller, N., Cai, Z., Lian, R., Huang, D., Yu, G.: Cognitive synergy between procedural and declarative learning in the control of animated and robotic agents using the opencogprime agi architecture. In: Proceedings of AAAI-11. (2011)
12. Goertzel, B., Et Al, C.P.: An integrative methodology for teaching embodied non-linguistic agents, applied to virtual animals in second life. In: Proc.of the First Conf. on AGI, IOS Press (2008)
13. Goertzel, B., Pitt, J., Cai, Z., Wigmore, J., Huang, D., Geisweiller, N., Lian, R., Yu, G.: Integrative general intelligence for controlling game ai in a minecraft-like environment. In: Proc. of BICA 2011. (2011)
14. Goertzel, B., Pinto, H., Pennachin, C., Goertzel, I.F.: Using dependency parsing and probabilistic inference to extract relationships between genes, proteins and malignancies implicit among multiple biomedical research abstracts. In: Proc. of Bio-NLP 2006. (2006)
15. Arel, I., Rose, D., Karnowski, T.: A deep learning architecture comprising homogeneous cortical circuits for scalable spatiotemporal pattern inference. NIPS 2009 Workshop on Deep Learning for Speech Recognition and Related Applications (2009)
16. Arel, I., Rose, D., Coop, R.: Destin: A scalable deep learning architecture with application to high-dimensional robust pattern recognition. Proc. AAAI Workshop on Biologically Inspired Cognitive Architectures (2009)
17. Hawkins, J., Blakeslee, S.: On Intelligence. Brown Walker (2006)
18. George, D., Hawkins, J.: Towards a mathematical theory of cortical micro-circuits. PLoS Comput Biol 5 (2009)
19. Tarifi, M., Sitharam, M., Ho, J.: Learning hierarchical sparse representations using iterative dictionary learning and dimension reduction. In: Proc. of BICA 2011. (2011)
20. Bundzel, Hashimoto: Object identification in dynamic images based on the memory-prediction theory of brain function. Journal of Intelligent Learning Systems and Applications **2-4** (2010)
21. Goertzel, B.: Integrating a compositional spatiotemporal deep learning network with symbolic representation/reasoning within an integrative cognitive architecture via an intermediary semantic network. In: Proceedings of AAAI Symposium on Cognitive Systems,. (2011)
22. Karnowski, T., Arel, I., Rose, D.: Deep spatiotemporal feature learning with application to image classification. In: The 9th International Conference on Machine Learning and Applications (ICMLA'10). (2010)
23. Lee, S.H., Kim, J., Park, F.C., Kim, M., Bobrow, J.E.: Newton-type algorithms for dynamics-based robot movement optimization. IEEE Transactions on Robotics **21**(4) (2005) 657–667
24. Yeo, S., Kim, J., Lee, S.H., Park, F.C., Park, W., Kim, J., Park, C., Yeo, I.: A modular object-oriented framework for hierarchical multi-resolution robot simulation. Robotica **22**(2) (2004) 141–154
25. Goertzel, B.: Modifying the destin perception architecture to enable representationally transparent deep learning. (http://wp.goertzel.org/?p=404)